

CHAPLIN - Complex Harmonic Polylogarithms in Fortran

Stephan Buehler^a, Claude Duhr^b,

^a*Institut für theoretische Physik, ETH Zürich,
Wolfgang-Paulistr. 27, CH-8093, Switzerland
Email: buehler@itp.phys.ethz.ch*

^b*Institute for Particle Physics Phenomenology, University of Durham,
Durham, DH1 3LE, United Kingdom,
Email: claudе.duhr@durham.ac.uk*

Abstract

We present a new Fortran library to evaluate all harmonic polylogarithms up to weight four numerically for any complex argument. The algorithm is based on a reduction of harmonic polylogarithms up to weight four to a minimal set of basis functions that are computed numerically using series expansions allowing for fast and reliable numerical results.

PACS: 12.38.Bx, Perturbative calculations

Key words: Harmonic polylogarithms, Fortran, loop computations.

PROGRAM SUMMARY

Manuscript Title: CHAPLIN - Complex Harmonic Polylogarithms in Fortran

Authors: Stephan Buehler, Claude Duhr

Program Title: Chaplin

Journal Reference:

Catalogue identifier:

Licensing provisions:

Programming language: Fortran 77

Computer: Computing systems on which Fortran 77 compilers are available.

Operating system: Operating systems on which Fortran 77 compilers are available.

Keywords: Harmonic polylogarithms, Fortran, loop computations.

PACS: 12.38.Bx, Perturbative calculations .

Classification: 11.1 General, High Energy Physics and Computing

Nature of problem: Numerical evaluation of harmonic polylogarithms.

Solution method: Inside the unit circle: series expansion. Outside the unit circle: inversion relations.

Restrictions: Only harmonic polylogarithms up to weight four are supported.

Unusual features: Allows to evaluate HPL's numerically for any point in the complex plane.

Running time: Depending on the weight vector and argument of the HPL, between 0.2 and 400 μ s.

1 Introduction

Feynman integrals in perturbative quantum field theory are generically expressed in terms of the classical polylogarithm functions $\text{Li}_n(z)$ and the Nielsen polylogarithms $S_{n,p}(z)$ [1]. In the late nineties, it was realized that these classes of functions are too restricted when going beyond one-loop level in the perturbative expansion, where new functions appear that can no longer be expressed in terms of the classical polylogarithm functions. While a completely generic generalization of polylogarithms has been studied in the mathematical literature (going under the name of multiple polylogarithms [2,3]), it is mostly only a specific subset of multiple polylogarithms, the so-called harmonic polylogarithms [4] and their two-dimensional and cyclotomic generalizations [5,6], that make their appearance in the theoretical predictions of physical quantities beyond leading-order.

In this paper we concentrate exclusively on harmonic polylogarithms (HPL's) up to weight four, which since their introduction have found many applications in computations up to two-loop order in the perturbative expansion, *e.g.*, [7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29]. In order to confront the theoretical next-to-next-to-leading order (NNLO) predictions to experiment, it is mandatory to be able to evaluate HPL's numerically in a fast and accurate way. The requirements to such a numerical code are twofold: first, the evaluation should be fast, because the use of NNLO matrix elements in Monte Carlo integration codes may require thousands, if not millions, of function calls. Second, it is desirable to be able to compute HPL's for arbitrary complex arguments, which appear for example when the complex mass scheme is employed or for certain kinematic configurations in loop calculations involving massive particles [14,29]. In the last decade, various codes have been developed to evaluate HPL's numerically. While the code `hpllog` [30], written in `Fortran`, is restricted to the evaluation of HPL's up to weight four and for real values of the arguments, the code `HPL (MATHEMATICA)` [31,32] and the implementation of the harmonic polylogarithms into the `GiNaC` framework (`C++`) [33] are generic and allow to evaluate in principle any harmonic polylogarithm with arbitrary precision for any complex argument.

The focus of this paper is the `CHAPLIN` (Complex HArmonic PolyLogarithms In `fortraN`) library, a new `Fortran` code that allows to evaluate numerically all harmonic polylogarithms up to weight four for arbitrary complex arguments. While `CHAPLIN` is similar in spirit to the aforementioned codes, its main advantages lie in speed, through the use of `Fortran` as a programming language, and in its capability to compute HPL's numerically for any point in the complex plane. `CHAPLIN` reduces each of the 120 HPL's up to weight four to a set of 32 basis functions [36], which are entirely expressed through

only two new functions of weight four besides the classical polylogarithms. The basis functions are then mapped to the interior of the unit circle, where they are computed numerically using suitably chosen series expansions that allow to obtain a fast numerical convergence.

The paper is organized as follows: In Section 2 we give a short review of harmonic polylogarithms and of their main algebraic and analytic properties. In Section 3 we review the reduction of all HPL's up to weight four to the set of basis functions introduced in Ref. [36]. The series expansions used by CHAPLIN to compute the basis functions numerically are derived in Section 4, while the CHAPLIN library itself, together with comparisons to `hpl`, `HPL` and `GiNaC`, is presented in Section 5.

2 Short review of harmonic polylogarithms

In this section we give a short review of harmonic polylogarithms (HPL's), as they are at the heart of the CHAPLIN library. HPL's are defined recursively via the iterated integrals [4]

$$H(a_1, \dots, a_n; z) = \int_0^z dt f(a_1; t) H(a_2, \dots, a_n; t), \quad (2.1)$$

where $a_i \in \{-1, 0, 1\}$ and

$$f(-1; z) = \frac{1}{1+z}, \quad f(0; z) = \frac{1}{z}, \quad f(1; z) = \frac{1}{1-z}, \quad (2.2)$$

and where we defined $H(; z) = 1$. If all the a_i are simultaneously zero, the integral (2.1) is divergent, and so in this case we use the definition,

$$H(\vec{0}_n; z) = \frac{1}{n!} \log^n z, \quad (2.3)$$

where we used the obvious vector notation $\vec{a}_n = (\underbrace{a, \dots, a}_{n \text{ times}})$. Note that the number n of indices is usually referred to as the *weight* of the HPL. Harmonic polylogarithms are a generalization of the classical polylogarithm functions, defined recursively by,

$$\text{Li}_n(z) = \int_0^z \frac{dt}{t} \text{Li}_{n-1}(t) \quad \text{and} \quad \text{Li}_1(z) = -\log(1-z). \quad (2.4)$$

Iterated integrals are well-known to form a shuffle algebra, and so in particular we can express a product of two harmonic polylogarithms of weight n_1 and n_2

as a linear combination of HPL's of weight $n_1 + n_2$,

$$H(\vec{a}_1; z) H(\vec{a}_2; z) = \sum_{\vec{a} = \vec{a}_1 \amalg \vec{a}_2} H(\vec{a}; z), \quad (2.5)$$

where $\vec{a}_1 \amalg \vec{a}_2$ denotes the shuffle of the two weight vectors \vec{a}_1 and \vec{a}_2 , *i.e.*, all possible concatenations of \vec{a}_1 and \vec{a}_2 in which relative orderings of \vec{a}_1 and \vec{a}_2 are preserved.

Up to weight three, HPL's are known to be expressible through ordinary logarithms and the classical polylogarithms Li_n only. Starting from weight four, not all HPL's can be expressed in terms of classical polylogarithms, and genuine new functions appear. For special classes of HPL's, however, it is possible to find closed expressions in terms of other functions. An example of this was already given in Eq. (2.3). Moreover, we have,

$$H(\vec{s}_n; z) = \frac{(-s)^n}{n!} \log^n(1 - s z) \quad \text{and} \quad H(\vec{0}_{n-1}, 1; z) = \text{Li}_n(z), \quad (2.6)$$

i.e., harmonic polylogarithms contain the classical polylogarithms as special cases.

Apart from the shuffle relation (2.5), HPL's satisfy various intricate functional equations, relating HPL's with different arguments among each other. As an example, the functional equation relating HPL's with opposite arguments reads,

$$H(\vec{a}; -z) = (-1)^p H(-\vec{a}; z), \quad (2.7)$$

where p denotes the number elements in \vec{a} equal to ± 1 . Furthermore, it is always possible to express harmonic polylogarithms of the form $H(\vec{a}; 1/z)$ as a linear combination of HPL's of the form $H(\vec{a}; z)$. This allows in particular to analytically continue the harmonic polylogarithms outside the unit disc, a property that will be used later in the numerical implementation of the HPL's into the CHAPLIN library. Additional relations among HPL's with related arguments have been presented in Ref. [4,31].

Let us conclude this section by discussing some special values of the argument z for which the HPL's can be expressed in terms of known transcendental numbers. First, it is easy to see that, unless $\vec{a} = \vec{0}_n$, all HPL's vanish for $z = 0$. Second, if the argument z of a harmonic polylogarithm is ± 1 , then it can be expressed in terms of so-called *colored multiple zeta values*¹ (CMZV's),

$$\begin{aligned} & H(\underbrace{0, \dots, 0}_{m_1-1}, s_1, \dots, \underbrace{0, \dots, 0}_{m_k-1}, s_k; 1) \\ &= (-1)^{m+p} \zeta(m_1, \dots, m_k; s_1, s_2/s_1, \dots, s_k/s_{k-1}), \end{aligned} \quad (2.8)$$

¹ Also known as *alternating multiple zeta values* or *Euler-Zagier sums*.

with $s_i = \pm 1$ and $m = m_1 + \dots + m_k$, and p denotes the number of elements in $\{s_i\}$ equal to $+1$. The CMZV's are defined by nested sums,

$$\zeta(m_1, \dots, m_k; \sigma_1, \dots, \sigma_k) = \sum_{0 < n_1 < n_2 < \dots < n_k} \frac{\sigma_1^{n_1} \sigma_2^{n_2} \dots \sigma_k^{n_k}}{n_1^{m_1} n_2^{m_2} \dots n_k^{m_k}}, \quad (2.9)$$

with $\sigma_i = \pm 1$. CMZV's are convergent if and only if $(m_1, \sigma_1) \neq (1, 1)$. It follows then immediately that HPL's of the form $H(\pm 1, \vec{a}; \pm 1)$ are in general divergent².

3 Reduction to basis functions

From the previous section it is clear that many HPL's are not independent functions, but they are related among themselves by various intricate relations. In order to achieve an efficient numerical implementation, it is desirable to have as few independent functions as possible, *i.e.*, we would like to resolve all the identities in order to arrive at a minimal set of functions, which are 'as simple as possible' and through which all other HPL's can be expressed.

All the functional equations among harmonic polylogarithms (or, more generically, among multiple polylogarithms) can be resolved through the so-called symbol calculus. At the heart of the symbol calculus is the so-called *symbol map* [34], a linear map that associates to an HPL of weight n a tensor of rank n . As an example, the tensor associated to the classical polylogarithm $\text{Li}_n(z) = H(\vec{0}_{n-1}, 1; z)$ reads,

$$\mathcal{S}(\text{Li}_n(z)) = -(1-z) \otimes \underbrace{z \otimes \dots \otimes z}_{(n-1) \text{ times}}. \quad (3.1)$$

Furthermore, the symbol maps products that appear inside the tensor product to a sum of tensors,

$$\dots \otimes (X \cdot Y) \otimes \dots = \dots \otimes X \otimes \dots + \dots \otimes Y \otimes \dots \quad (3.2)$$

It is conjectured that all the functional identities among (multiple) polylogarithms are mapped under the symbol map \mathcal{S} to algebraic relations among the tensors. Hence, the symbol calculus provides an effective way to resolve all the functional equations among (a certain class of) multiple polylogarithms.

In Ref. [36], the symbol map was used to obtain a set of basis functions through which all HPL's up to weight four can be expressed. The basis functions obtained in Ref. [36] read,

² In some cases the divergence can be tamed, *e.g.*, $\lim_{z \rightarrow 0} H(1, 0; z) = -\zeta_2$.

- for weight one,

$$\mathcal{B}_1^{(1)}(z) = \log z, \quad \mathcal{B}_1^{(2)}(z) = \log(1 - z), \quad \mathcal{B}_1^{(3)}(z) = \log(1 + z), \quad (3.3)$$

- for weight two,

$$\mathcal{B}_2^{(1)}(z) = \text{Li}_2(z), \quad \mathcal{B}_2^{(2)}(z) = \text{Li}_2(-z), \quad \mathcal{B}_2^{(3)}(z) = \text{Li}_2\left(\frac{1-z}{2}\right), \quad (3.4)$$

- for weight three,

$$\begin{aligned} \mathcal{B}_3^{(1)}(z) &= \text{Li}_3(z), & \mathcal{B}_3^{(2)}(z) &= \text{Li}_3(-z), & \mathcal{B}_3^{(3)}(z) &= \text{Li}_3(1-z), \\ \mathcal{B}_3^{(4)}(z) &= \text{Li}_3\left(\frac{1}{1+z}\right), & \mathcal{B}_3^{(5)}(z) &= \text{Li}_3\left(\frac{1+z}{2}\right), & \mathcal{B}_3^{(6)}(z) &= \text{Li}_3\left(\frac{1-z}{2}\right), \\ \mathcal{B}_3^{(7)}(z) &= \text{Li}_3\left(\frac{1-z}{1+z}\right), & \mathcal{B}_3^{(8)}(z) &= \text{Li}_3\left(\frac{2z}{z-1}\right), \end{aligned} \quad (3.5)$$

- for weight four,

$$\begin{aligned} \mathcal{B}_4^{(1)}(z) &= \text{Li}_4(z), & \mathcal{B}_4^{(2)}(z) &= \text{Li}_4(-z), \\ \mathcal{B}_4^{(3)}(z) &= \text{Li}_4(1-z), & \mathcal{B}_4^{(4)}(z) &= \text{Li}_4\left(\frac{1}{1+z}\right), \\ \mathcal{B}_4^{(5)}(z) &= \text{Li}_4\left(\frac{z}{z-1}\right), & \mathcal{B}_4^{(6)}(z) &= \text{Li}_4\left(\frac{z}{z+1}\right), \\ \mathcal{B}_4^{(7)}(z) &= \text{Li}_4\left(\frac{1+z}{2}\right), & \mathcal{B}_4^{(8)}(z) &= \text{Li}_4\left(\frac{1-z}{2}\right), \\ \mathcal{B}_4^{(9)}(z) &= \text{Li}_4\left(\frac{1-z}{1+z}\right), & \mathcal{B}_4^{(10)}(z) &= \text{Li}_4\left(\frac{z-1}{z+1}\right), \\ \mathcal{B}_4^{(11)}(z) &= \text{Li}_4\left(\frac{2z}{z+1}\right), & \mathcal{B}_4^{(12)}(z) &= \text{Li}_4\left(\frac{2z}{z-1}\right), \\ \mathcal{B}_4^{(13)}(z) &= \text{Li}_4(1-z^2), & \mathcal{B}_4^{(14)}(z) &= \text{Li}_4\left(\frac{z^2}{z^2-1}\right), \\ \mathcal{B}_4^{(15)}(z) &= \text{Li}_4\left(\frac{4z}{(z+1)^2}\right). \end{aligned} \quad (3.6)$$

All harmonic polylogarithms up to weight three can be expressed through the basis functions in Eq. (3.3 - 3.6). Starting from weight four, we need to extend the set of functions by adjoining three new elements to the basis,

$$\begin{aligned} \mathcal{B}_4^{(16)}(z) &= \text{Li}_{2,2}(-1, z), & \mathcal{B}_4^{(17)}(z) &= \text{Li}_{2,2}\left(\frac{1}{2}, \frac{2z}{z+1}\right), \\ \mathcal{B}_4^{(18)}(z) &= \text{Li}_{2,2}\left(\frac{1}{2}, \frac{2z}{z-1}\right), \end{aligned} \quad (3.7)$$

where $\text{Li}_{2,2}$ denotes a two-variable multiple polylogarithm that cannot be ex-

pressed through classical polylogarithms only,

$$\text{Li}_{2,2}(z_1, z_2) = \sum_{n_1=1}^{\infty} \sum_{n_2=1}^{n_1-1} \frac{z_1^{n_1}}{n_1^2} \frac{z_2^{n_2}}{n_2^2}. \quad (3.8)$$

For practical purposes we find it more convenient to use a different set of multiple polylogarithms as basis functions than the one used in Ref. [36]. More specifically, we find it more convenient to perform a change of basis and replace the functions $\mathcal{B}_4^{(i)}(z)$, for $i = 16, 17, 18$, by the functions $\tilde{\mathcal{B}}_4^{(i)}(z)$, which are directly expressed as HPL's,

$$\begin{aligned} \tilde{\mathcal{B}}_4^{(16)}(z) &= H(0, 1, 0, -1; z) = \mathcal{B}_4^{(16)}(-z), \\ \tilde{\mathcal{B}}_4^{(17)}(z) &= H(0, 1, 1, -1; z), \\ \tilde{\mathcal{B}}_4^{(18)}(z) &= H(0, 1, 1, -1; -z). \end{aligned} \quad (3.9)$$

The set of the 32 functions $\mathcal{B}_i^{(j)}(z)$ defines a basis through which all HPL's up to weight four can be expressed. As a consequence, any numerical code to evaluate this set of basis functions will automatically be able to evaluate all 120 HPL's up to weight four. Furthermore, as the basis functions only involve the two genuine multiple polylogarithms $H(0, 1, 0, -1; z)$ and $H(0, 1, 1, -1; z)$ besides the ordinary logarithms and the classical polylogarithms $\text{Li}_2(z)$, $\text{Li}_3(z)$ and $\text{Li}_4(z)$, it is enough to have numerical routines for these latter functions. In this way we can reduce the problem of evaluating the 120 HPL's up to weight four to only a handful of non-trivial numerical routines. In the CHAPLIN library, these routines consist of series expansions for the aforementioned functions that will be described in the next section.

Let us conclude this section by reviewing some of the properties of the basis functions $\mathcal{B}_i^{(j)}(z)$ derived in Ref. [36]. First, it is easy to check that all the basis functions are real for $z \in [0, 1]$. However, we stress that the expressions in Eq. (3.3 - 3.9) are strictly valid only for $z \in [0, 1]$. While most of the expressions are valid everywhere throughout the unit disc, the analytic form of $\mathcal{B}_4^{(13)}(z)$ valid on the whole interior of the disc reads [36],

$$\begin{aligned} &\mathcal{B}_4^{(13)}(z) \\ &= \begin{cases} \text{Li}_4(1 - z^2), & \text{if } \text{Re}(z) > 0 \quad \text{or } (\text{Re}(z) = 0 \text{ and } \text{Im}(z) \geq 0), \\ \text{Li}_4(1 - z^2) - \frac{i\pi}{3} \sigma(z) \log^3(1 - z^2), & \text{otherwise,} \end{cases} \end{aligned} \quad (3.10)$$

where $\sigma(z) = \text{sign}(\text{Im}(z))$. Second, since it is our goal to build a numerical code to evaluate HPL's for arbitrary complex arguments, we need to analytically continue the basis functions outside the unit disc. In Ref. [36] inversion

relations of the form

$$\mathcal{B}_j^{(i)}(z) = \sum_{k,l} c_{ijkl} \mathcal{B}_k^{(l)} \left(\frac{1}{z} \right) + \text{products of lower weight.} \quad (3.11)$$

were derived that can be used for this purpose. Finally, we note that there is a subtlety in the basis function $\mathcal{B}_4^{(15)}(z)$ when going from the interior to the exterior of the unit disc because $\mathcal{B}_4^{(15)}(z)$ has a branch cut along the unit circle in the complex z -plane. In Ref. [36] it was shown that if we want $\mathcal{B}_4^{(15)}(z)$ to be continuous and real for $z \in [0, 1]$, we need to choose the following prescription for $|z| = 1$,

$$\mathcal{B}_4^{(15)}(z) = \text{Li}_4 \left(\frac{4z}{(1+z)^2} + i\sigma(z)\varepsilon \right). \quad (3.12)$$

4 Numerical evaluation of the basis functions

4.1 Notations and conventions

In the previous section we introduced a set of basis functions through which every HPL up to weight four can be expressed. The basis has the property that it only involves two types of new functions, besides the ordinary logarithm and the classical polylogarithms. These two new functions can be chosen to correspond to the two harmonic polylogarithms $H(0, 1, 0, -1; z)$ and $H(0, 1, 1, -1; z)$. In this section we present series expansions of these functions which are used inside CHAPLIN to evaluate the basis functions.

Let us start by introducing some notations and conventions. As we will deal with series expansions, we define some operations on the coefficients of the series, *i.e.*, on sequences of complex numbers. For two sequences of complex numbers a_n and b_n , $n \in \mathbb{N}$, we define their convolution product as the sequence $(a * b)_n$ defined by

$$(a * b)_n = \sum_{k=0}^n \binom{n}{k} a_k b_{n-k}. \quad (4.1)$$

It is easy to check by manipulating the sum that this operation is associative, commutative and has the sequence $\varepsilon_n = \delta_{0,n}$, where $\delta_{i,j}$ is the Kronecker symbol, as a neutral element,

$$a * (b * c) = (a * b) * c, \quad a * b = b * a, \quad a * \varepsilon = \varepsilon * a = a. \quad (4.2)$$

The fact that ε_n is a neutral element is obvious, and the commutativity follows immediately from changing the summation variable from k to $n - k$. Associativity is less obvious, and is proved in Appendix A. Furthermore, this operation

is compatible with the usual termwise addition and scalar multiplication of sequences,

$$a * (b + c) = a * b + a * c, \quad (\kappa \cdot a) * b = \kappa \cdot (a * b), \quad (4.3)$$

where a , b and c are sequences of complex numbers and κ is a constant complex number. The convolution product allows us to write the coefficients that appear in the product of two power series as the convolution of the coefficients of the individual factors, *e.g.*,

$$\left(\sum_{m=0}^{\infty} \frac{a_m}{m!} x^m \right) \left(\sum_{n=0}^{\infty} \frac{b_n}{n!} x^n \right) = \sum_{N=0}^{\infty} \frac{(a * b)_N}{N!} x^N. \quad (4.4)$$

Finally, for later convenience, we define for a given sequence a_n of complex numbers the three new sequences \mathring{a}_n , \bar{a}_n and $s(a)_n$ by

$$\mathring{a}_n = \frac{a_n}{n+1}, \quad \bar{a}_n = (-1)^n a_n, \quad s(a)_n = \begin{cases} a_{n-1}, & \text{if } n \geq 1 \\ 0, & \text{otherwise} \end{cases}. \quad (4.5)$$

The bar-operation allows us to define the coefficients of the series expansion of $f(-z)$ in terms of the coefficients of the series expansion of $f(z)$. More precisely, the two series expansions are related by

$$f(z) = \sum_{n=0}^{\infty} f_n z^n \quad \text{and} \quad f(-z) = \sum_{n=0}^{\infty} \bar{f}_n z^n. \quad (4.6)$$

We also define the composition of the ‘ \circ ’-operation and the convolution and shift operations,

$$(a \circledast b)_n = (a * b)_n^\circ = \frac{(a * b)_n}{n+1} \quad \text{and} \quad \mathring{s}(a)_n = \frac{s(a)_n}{n+1}. \quad (4.7)$$

Note that the \circledast -operation is commutative, but not associative. For later convenience we introduce the following convention,

$$a \circledast b \circledast c \equiv a \circledast (b \circledast c). \quad (4.8)$$

The operations on sequences of complex numbers we just defined allow us to write the coefficients appearing in the series expansion of the basis functions in a compact closed form. The sequences of complex numbers that appear inside these closed-form expressions are well-known sequences of (rational) numbers which we recall in the following.

(1) **(Shifted) ζ values:**

$$\zeta_n^{(k)} = \begin{cases} \zeta_{k-n}, & \text{if } k - n \neq 1, \\ H_{k-1}, & \text{if } k - n = 1, \end{cases} \quad (4.9)$$

where $\zeta_m \equiv \zeta(m)$ denotes the Riemann ζ function and H_m the m -th harmonic number,

$$\zeta(z) = \sum_{n=1}^{\infty} \frac{1}{n^z} \quad \text{and} \quad H_m = \sum_{n=1}^m \frac{1}{n}. \quad (4.10)$$

Note that $\zeta_n^{(k)}$ is rational for $n \geq k$ and transcendental of weight $k - n$ otherwise.

- (2) **Bernoulli numbers:** The Bernoulli numbers B_n are defined through the generating series

$$\frac{z}{e^z - 1} = \sum_{n=0}^{\infty} B_n \frac{z^n}{n!}. \quad (4.11)$$

Note that $B_{2n+1} = 0$, $\forall n \in \mathbb{N}$. The Bernoulli numbers are related to ζ values via the formula,

$$\zeta_0 = B_1 = -\frac{1}{2} \quad \text{and} \quad \zeta_{-n} = -\frac{B_{n+1}}{n+1} \quad \text{for } n \geq 1. \quad (4.12)$$

- (3) **Genocchi numbers:** The Genocchi numbers are defined through the generating series

$$\frac{2t}{1 + e^t} = \sum_{n=0}^{\infty} \frac{G_n}{n!} t^n. \quad (4.13)$$

They are related to the Bernoulli numbers via

$$G_n = 2(1 - 2^n) B_n. \quad (4.14)$$

Note that $G_0 = G_{2n+1} = 0$.

- (4) **Polylogarithms in half-integer values:** The series expansion of the basis functions also involve the following sequences of numbers

$$\ell_n = (-1)^n \text{Li}_{-n} \left(\frac{1}{2} \right). \quad (4.15)$$

The sequence ℓ_n admits the generating function,

$$\frac{1}{2e^z - 1} = \sum_{n=0}^{\infty} \ell_n \frac{z^n}{n!}. \quad (4.16)$$

4.2 Numerical evaluation of classical polylogarithms

In this section we discuss series expansions of classical polylogarithms that can be used to obtain reliable numerical results for these functions (at least in some regions of the complex plane). In particular, truncated versions of these series are used by CHAPLIN to evaluate the classical polylogarithms. As the goal of CHAPLIN is to provide an efficient way to evaluate harmonic

polylogarithms for arbitrary complex arguments, we divide the problem into two regions: for complex numbers z with $|z| \leq 1$ we use the series expansions to evaluate the basis functions numerically, whereas points with $|z| > 1$ are mapped back into the interior of the unit disc using the inversion formulæ for the classical polylogarithms. Hence, from here on we will only concentrate on complex number z with $|z| \leq 1$.

Classical polylogarithms can be expanded into a power series around $z = 0$,

$$\text{Li}_n(z) = \sum_{k=1}^{\infty} \frac{z^k}{k^n}. \quad (4.17)$$

Even though this series is convergent for $|z| < 1$, the convergence is rather slow. A faster convergence can be achieved by using the so-called Bernoulli substitution [33], which consists in expanding $\text{Li}_n(z)$ into a series in $\log(1 - z)$. While this expansion converges much faster for $|z| \ll 1$ than the Taylor expansion (4.17), it fails to produce reliable results when z approaches 1. In Ref. [35] an alternative expansion of the classical polylogarithms into a series in $\log z$ was derived. In this case the convergence is fast inside an annulus around $z = 0$, but fails to converge for $|z| \ll 1$. The strategy seems thus clear: we can split the interior of the unit disc into two distinct regions, and in each region one of the two series expansions converges quickly. Similar expansions can also be derived for the two remaining basis functions, $H(0, 1, 0, -1; z)$ and $H(0, 1, 1, -1; z)$ (and in principle for every HPL) and are discussed in the rest of this section. We start by deriving in detail the expansions of the dilogarithm, because even though these results are well-known, the techniques used in the derivation will be the starting point for the higher-weight cases.

Series expansions of $\text{Li}_2(z)$. Let us start by deriving the expansion of $\text{Li}_2(z)$ into a series in $\log(1 - z)$. Letting $x = -\log(1 - z)$, this is equivalent to finding the Taylor series expansion of the function $\text{Li}_2(1 - e^{-x})$. We start from the integral representation (2.4) and we get,

$$\text{Li}_2(1 - e^{-x}) = \int_0^{1-e^{-x}} \frac{dt}{t} \text{Li}_1(t) = - \int_0^{1-e^{-x}} \frac{dt}{t} \log(1 - t). \quad (4.18)$$

Performing the change of variables $t = 1 - e^{-t'}$ and using Eq. (4.11), we obtain,

$$\text{Li}_2(1 - e^{-x}) = - \int_0^x \frac{e^{-t'} dt'}{1 - e^{-t'}} (-t') = \int_0^x dt' \frac{t'}{e^{t'} - 1} = \sum_{k=0}^{\infty} \frac{B_k}{(k+1)!} x^{k+1}, \quad (4.19)$$

or equivalently in terms of the original variable z ,

$$\text{Li}_2(z) = \sum_{k=0}^{\infty} \frac{B_k}{(k+1)!} (-\log(1 - z))^{k+1}. \quad (4.20)$$

The series expansion (4.20) converges rather quickly inside a disc around $z = 0$ of radius $R < 1$ (the precise value of R used by CHAPLIN will be given in the next section). In the remaining annulus $R < |z| < 1$ the dilogarithm admits a series expansion in $\log z$ [35],

$$\text{Li}_2(z) = -\log z \log(-\log z) + \sum_{k=0}^{\infty} \frac{\zeta_k^{(2)}}{k!} \log^k z. \quad (4.21)$$

Let us sketch the derivation of Eq. (4.21). Letting $x = \log z$, we start from the integral representation of the dilogarithm and perform the change of variable $t = e^{t'}$,

$$\text{Li}_2(e^x) = \zeta_2 + \int_1^{e^x} \frac{dt}{t} \text{Li}_1(t) = \zeta_2 + \int_0^x dt' \text{Li}_1(e^{t'}). \quad (4.22)$$

In order to proceed, we need the Taylor expansion of $\text{Li}_1(e^x) = -\log(1 - e^x)$. Using the integral representation of Li_1 as well as Eq. (4.11), we obtain,

$$\begin{aligned} \text{Li}_1(e^x) &= \int_0^{e^x} \frac{dt}{1-t} = \lim_{\varepsilon \rightarrow 0} \left[-\log(1 - e^\varepsilon) - \int_\varepsilon^x \frac{dt'}{t'} \frac{(-t')}{e^{-t'} - 1} \right] \\ &= \lim_{\varepsilon \rightarrow 0} \left[-\log(1 - e^\varepsilon) - \int_\varepsilon^x \frac{dt'}{t'} - \sum_{n=1}^{\infty} \frac{B_n}{n!} \int_\varepsilon^x dt' (-t')^{n-1} \right]. \end{aligned} \quad (4.23)$$

The last term in Eq. (4.23) is finite, whereas the logarithmic divergences cancel between the first two terms,

$$\begin{aligned} \lim_{\varepsilon \rightarrow 0} \left[-\log(1 - e^\varepsilon) - \int_\varepsilon^x \frac{dt'}{t'} \right] &= \lim_{\varepsilon \rightarrow 0} \left[-\log(-\varepsilon + \mathcal{O}(\varepsilon^2)) - \log(-x) + \log(-\varepsilon) \right] \\ &= \lim_{\varepsilon \rightarrow 0} \left[-\log(1 + \mathcal{O}(\varepsilon)) - \log(-x) \right] = -\log(-x). \end{aligned} \quad (4.24)$$

Hence, using Eq. (4.12) and the fact that $\zeta_{-n} = 0$ for even n , we get,

$$\begin{aligned} \text{Li}_1(e^x) &= -\log(-x) - \sum_{n=1}^{\infty} \frac{B_n}{n!} \frac{(-x)^n}{n} = -\log(-x) + \zeta_0 x + \sum_{n=1}^{\infty} \frac{\zeta_{-n}}{(n+1)!} (-x)^{n+1} \\ &= -\log(-x) + \sum_{n=0}^{\infty} \frac{\zeta_{-n}}{(n+1)!} x^{n+1}. \end{aligned} \quad (4.25)$$

Inserting Eq. (4.25) into Eq. (4.22) and integrating term by term immediately reproduces Eq. (4.21) (after identification $x = \log z$). Note the appearance of the nested logarithm, $\log(-\log z)$ in Eq. (4.21), which seems to be divergent for z close to 1. It is however easy to check that

$$\lim_{z \rightarrow 1^-} \log z \log(-\log z) = 0, \quad (4.26)$$

and so the whole expression is well behaved.

Series expansions of $\text{Li}_n(z)$, $n > 2$. The derivations of the series expansions of $\text{Li}_2(z)$ presented in the previous section are by no means restricted to the weight two case, but we can repeat exactly the same steps for classical polylogarithms of arbitrary weight. In Ref. [35] the following more general version of Eq. (4.21) was proven,

$$\text{Li}_n(z) = -\frac{1}{(n-1)!} \log^{n-1} z \log(-\log z) + \sum_{k=0}^{\infty} \frac{\zeta_k^{(n)}}{k!} \log^k z. \quad (4.27)$$

The proof goes by recursion in the weight, and we refer to Ref. [35] for more details.

The generalization of Eq. (4.20) to arbitrary weight is simply given by,

$$\text{Li}_n(z) = \sum_{k=0}^{\infty} \frac{1}{(k+1)!} [B * \underbrace{(B \circledast \dots \circledast B \circledast \mathring{B})}_{n-2 \text{ times}}]_k (-\log(1-z))^{k+1}. \quad (4.28)$$

The proof goes by recursion in the weight. We have shown in the previous section that Eq. (4.28) is true for $n = 2$. If we assume in addition Eq. (4.28) true up to weight n , then we obtain,

$$\begin{aligned} \text{Li}_{n+1}(1-e^{-x}) &= \int_0^{1-e^{-x}} \frac{dt}{t} \text{Li}_n(t) \\ &= \sum_{l=0}^{\infty} \frac{1}{(l+1)!} [B * \underbrace{(B \circledast \dots \circledast B \circledast \mathring{B})}_{n-2 \text{ times}}]_l \int_0^x \frac{t' dt'}{e^{t'-1}} t'^l \\ &= \sum_{l=0}^{\infty} \frac{1}{l!} \underbrace{(B \circledast \dots \circledast B \circledast \mathring{B})_l}_{n-1 \text{ times}} \sum_{m=0}^{\infty} \frac{B_m}{m!} \frac{x^{l+m+1}}{l+m+1} \\ &= \sum_{k=0}^{\infty} \frac{1}{k!} [B * \underbrace{(B \circledast \dots \circledast B \circledast \mathring{B})}_{n-1 \text{ times}}]_k \frac{x^{k+1}}{k+1}. \end{aligned} \quad (4.29)$$

The series expansions in $\log z$ and $\log(1-z)$ for the classical polylogarithms are enough to evaluate all basis functions, except $\tilde{B}_4^{(i)}$, for $i \in \{16, 17, 18\}$, numerically in a fast and reliable way. For $\tilde{B}_4^{(i)}$, for $i \in \{16, 17, 18\}$ we need to extend these series expansions beyond the case of classical polylogarithms.

4.3 Series expansions of $H(0, 1, 0, -1; z)$ and $H(0, 1, 1, -1; z)$

In this section we present the analogues of the series expansions in $\log(1-z)$ and $\log z$ of the previous section for the three remaining basis functions that cannot be expressed in terms of classical polylogarithms only, namely $\tilde{B}_4^{(i)}$, for $i \in \{16, 17, 18\}$. As the derivation of the expansions follows exactly the same lines as for the classical polylogarithms, we content ourselves to present the results and refer to Appendix B for details.

Let us start by deriving the expansion of $\tilde{B}_4^{(i)}$, $i \in \{16, 17, 18\}$ into a series in $\log(1 - z)$. We can obtain these expansions as a corollary of a more general result: If a function $\mathcal{H}(z)$ admits a Taylor expansion of the form

$$\mathcal{H}(1 - e^{-x}) = \sum_{n=0}^{\infty} \frac{h_n}{(n+1)!} x^{n+1}, \quad (4.30)$$

then for $a \in \{-1, 0, 1\}$ the functions $\mathcal{H}_a(z)$ defined by

$$\mathcal{H}_a(z) = \int_0^z dt f(a; z) \mathcal{H}(t), \quad (4.31)$$

where $f(a; z)$ was defined in Eq. (2.2), admit the following Taylor expansions

$$\begin{aligned} \mathcal{H}_1(1 - e^{-x}) &= \sum_{n=0}^{\infty} \frac{s(h)_n}{(n+1)!} x^{n+1} = \sum_{n=0}^{\infty} \frac{h_n}{(n+2)!} x^{n+2}, \\ \mathcal{H}_0(1 - e^{-x}) &= \sum_{n=0}^{\infty} \frac{(B * \check{h})_n}{(n+1)!} x^{n+1}, \\ \mathcal{H}_{-1}(1 - e^{-x}) &= \sum_{n=0}^{\infty} \frac{(\ell * s(h))_n}{(n+1)!} x^{n+1}. \end{aligned} \quad (4.32)$$

To prove these identities we start from the integral representation (4.31) and, after having performed the change of variable $t = 1 - e^{-t'}$, we insert the series expansion for the integrand and integrate term by term, which yields immediately Eq. (4.32). Combined with the expansions for HPL's of weight one, Eq. (4.32) allows us in principle to recursively expand HPL's of arbitrary weight into a series in $\log(1 - z)$. A similar iterative procedure was already described in Ref. [33]. As the classical polylogarithms are just a special case of the harmonic polylogarithms, Eq. (4.28) can be seen as a special solution of Eq. (4.32) for $H(\vec{0}_{n-1}, 1; 1 - e^{-x})$. Moreover, we can easily read off from Eq. (4.32) the corresponding expansions of $H(0, 1, 0, -1; z)$ and $H(0, 1, 1, -1; z)$,

$$\begin{aligned} H(0, 1, 0, -1; z) &= \sum_{n=0}^{\infty} \frac{[B * \check{s}(B * \check{\ell})]_n}{(n+1)!} (-\log(1 - z))^{n+1}, \\ H(0, 1, 1, -1; z) &= \sum_{n=0}^{\infty} \frac{[B * \check{s}^2(\ell)]_n}{(n+1)!} (-\log(1 - z))^{n+1}, \end{aligned} \quad (4.33)$$

with $\check{s}^2(\ell)_n = \ell_{n-2}/(n+1)$. The expansions (4.33) are used inside `CHAPLIN` to evaluate the basis functions $\tilde{B}_4^{(i)}(z)$, $i \in \{16, 17, 18\}$ for z close to the origin. Similar to the case of the classical polylogarithms, these series converge rather slowly if z is close to 1. We therefore need additional expansions in $\log z$ that have a good convergence behavior in that region.

In an annulus inside the unit disc, the basis functions $\tilde{B}_4^{(i)}(z)$, $i \in \{16, 17, 18\}$ can be expanded into a series in $\log z$. As these basis functions are entirely

expressed through $H(0, 1, 0, -1; z)$ and $H(0, 1, 1, -1; z)$, it is enough to find the expansions for these cases. We obtain,

$$\begin{aligned}
H(0, 1, 0, -1; z) &= -4 \operatorname{Li}_4\left(\frac{1}{2}\right) - \frac{5}{2} \zeta_3 \log 2 + \frac{17\pi^4}{480} - \frac{1}{6} \log^4 2 \\
&+ \frac{\pi^2}{6} \log^2 2 - \frac{5}{8} \zeta_3 \log z + \frac{\pi^2}{6} \log 2 \log z + \frac{\pi^2}{12} \operatorname{Li}_2(z) \\
&+ \log 2 \log z \operatorname{Li}_2(z) - 2 \log 2 \operatorname{Li}_3(z) + \frac{1}{2} \sum_{n=0}^{\infty} \frac{(\overline{B} * \hat{\gamma})_n}{(n+2)!} \log^{n+2} z, \\
H(0, 1, 1, -1; z) &= \operatorname{Li}_4\left(\frac{1}{2}\right) - \frac{1}{8} \zeta_3 \log 2 + \frac{\pi^4}{720} + \frac{1}{24} \log^4 2 \\
&+ \left(\frac{1}{2} \log^2 2 - \frac{\pi^2}{12}\right) \operatorname{Li}_2(z) + \left(\frac{7}{8} \zeta_3 + \frac{1}{6} \log^3 2 - \frac{\pi^2}{12} \log 2\right) \log z \\
&- \frac{1}{2} \sum_{n=0}^{\infty} \frac{(\overline{B} * \beta)_n}{(n+1)!} \frac{1}{n} \log^{n+1} z,
\end{aligned} \tag{4.34}$$

with

$$\gamma_n = \frac{\overline{G}_n}{n} \quad \text{and} \quad \beta_n = \frac{(\overline{B} * \gamma)_n}{n}. \tag{4.35}$$

The proof of the Eq. (4.34) is sketched in Appendix B. A similar expansion in the particular case of $H(1, \vec{0}_n, 1; z)$ has already been considered in Ref. [37]. We stress that the relations (4.34) are only valid in the region $\operatorname{Re}(z) \geq 0$ and $R \leq |z| \leq 1$, for some R (we give a precise value for R in the next section when discussing the implementation of these expansions into CHAPLIN). For $\operatorname{Re}(z) < 0$, we proceed in following way,

- for $H(0, 1, 0, -1; z)$, we map the problem to $\operatorname{Re}(z) > 0$ via the functional equation

$$\begin{aligned}
H(0, 1, 0, -1; z) &= -H(0, -1, 0, 1; -z) \\
&= H(0, 1, 0, -1; -z) - \frac{1}{2} \operatorname{Li}_4(1 - z^2) + \frac{1}{2} \operatorname{Li}_4\left(\frac{z^2}{z^2 - 1}\right) \\
&- 2 \operatorname{Li}_4\left(\frac{1}{1 - z}\right) + 2 \operatorname{Li}_4(-z) + 2 \operatorname{Li}_4(z) - 2 \operatorname{Li}_4\left(\frac{z}{z + 1}\right) \\
&- 2 \operatorname{Li}_4\left(\frac{z}{z - 1}\right) + 2 \operatorname{Li}_4(1 + z) - 2 \operatorname{Li}_3(-z) \log(1 - z) \\
&- 2 \operatorname{Li}_3(z) \log(1 + z) - \frac{3}{2} \zeta_3 \log(1 - z) - \frac{3}{2} \zeta_3 \log(1 + z) \\
&- \frac{7}{48} \log^4(1 - z) - \frac{1}{16} \log^4(1 + z) + \frac{1}{6} \log(-z) \log^3(1 - z) \\
&+ \frac{1}{12} \log(1 + z) \log^3(1 - z) + \frac{1}{12} \log^3(1 + z) \log(1 - z)
\end{aligned} \tag{4.36}$$

$$\begin{aligned}
& + \frac{1}{6} \log(-z) \log^3(1+z) + \frac{1}{8} \log^2(1+z) \log^2(1-z) \\
& - \frac{1}{2} \log(-z) \log(1+z) \log^2(1-z) + \frac{5\pi^2}{24} \log^2(1-z) \\
& - \frac{1}{2} \log(-z) \log^2(1+z) \log(1-z) - \frac{\pi^2}{8} \log^2(1+z) \\
& - \text{Li}_2(-z) \text{Li}_2(z) + \frac{\pi^2}{12} \log(1+z) \log(1-z) + \frac{\pi^4}{180}.
\end{aligned} \tag{4.37}$$

- for $H(0, 1, 1, -1; z)$, we use the following expansion, valid for $\text{Re}(z) < 0$,

$$\begin{aligned}
H(0, 1, 1, -1; z) = & 2 \text{Li}_4\left(\frac{1}{2}\right) - \frac{7\pi^4}{360} + \frac{1}{12} \log^4 2 + \frac{\pi^2}{12} \log^2 2 \\
& + \left(\frac{1}{8} \zeta_3 - \frac{1}{6} \log^3 2 + \log 2\right) \log z - \left(\frac{\pi^2}{12} - \frac{1}{2} \log^2 2\right) \text{Li}_2(-z) \\
& - \frac{1}{4} \sum_{n=0}^{\infty} \frac{(\overline{G} * \bar{g}^{(1)})_n}{(n+1)!} \frac{1}{n} \log^{n+1}(-z) \left(\log(-\log(-z)) - \frac{1}{n+1} - \frac{1}{n}\right) \\
& + \frac{1}{4} \sum_{n=0}^{\infty} \frac{(\overline{G} * \bar{g}^{(2)})_n}{(n+1)!} \frac{1}{n} \log^{n+1}(-z) + \frac{1}{4} \sum_{n=0}^{\infty} \frac{[\overline{G} * (\overline{G} \circledast \xi)]_n}{(n+2)!} \log^{n+2}(-z),
\end{aligned} \tag{4.38}$$

with

$$g_n^{(i)} = \frac{G_n}{n^i} \quad \text{and} \quad \xi_n = \zeta_{-n}. \tag{4.39}$$

At this stage we have all the ingredients to evaluate all harmonic polylogarithms up to weight four numerically for arbitrary complex arguments. To this effect, we have implemented the decomposition of HPL's to the basis defined by Eq. (3.3 - 3.9) and the series expansions presented in this section into the CHAPLIN library, which we will present in the next section.

5 The Fortran library CHAPLIN

5.1 Installation

The CHAPLIN code is available as a `.tar` archive on the website <http://projects.hepforge.org/chaplin/>.

Once unpacked, the code can be compiled via

```
./configure
make install
```

As a result, both a static and a shared library are created in the directory `/usr/lib`, which may require root privileges. The directory which the library is installed in can be changed during configuration via

```
./configure --prefix="/path to Chaplin/"
```

This feature allows to create the library even without root privileges.

We advocate the use of the shared library when linking CHAPLIN to a **Fortran** code, such that the routines needed by the program will only be called during run-time. Static linking, on the contrary, puts all the CHAPLIN functions into the executable CHAPLIN is linked against, which might result in long compilation times and rather large executables.

5.2 *Running* CHAPLIN

Once compiled, the CHAPLIN library can be linked to other programs in the same way as any other library. This enables the user to call the numerical routines of CHAPLIN from within his/her own program. The function calls to the numerical routines of CHAPLIN are done via the following functions,

```
double complex HPL1(n1, z)
double complex HPL2(n1, n2, z)
double complex HPL3(n1, n2, n3, z)
double complex HPL4(n1, n2, n3, n4, z)
```

with $n_i \in \{-1, 0, 1\}$ and z being any **double complex** number. The return value is $H(\vec{n}; z)$, *i.e.*, the HPL with weight vector \vec{n} at the point z . Alternatively, in case only the real and/or imaginary parts of an HPL are needed, the user might find the following functions convenient,

```
double precision HPL2real(n1, n2, xr, xi)
double precision HPL2im(n1, n2, xr, xi)
```

and similarly for HPL3, HPL4. The variables **xr**, **xi** are **double precision** variables, denoting the real and imaginary parts of the argument of the HPL. Note that these functions are useful to call CHAPLIN from within a **C++** program, where complex numbers are not natively supported.

When one of the aforementioned functions is called, CHAPLIN starts by decomposing the corresponding HPL internally into the basis of 32 functions described in Section 3. In a second step, CHAPLIN proceeds to the numerical evaluation of the individual basis functions appearing in the decomposition. The numerical routines called to this effect are different depending on the

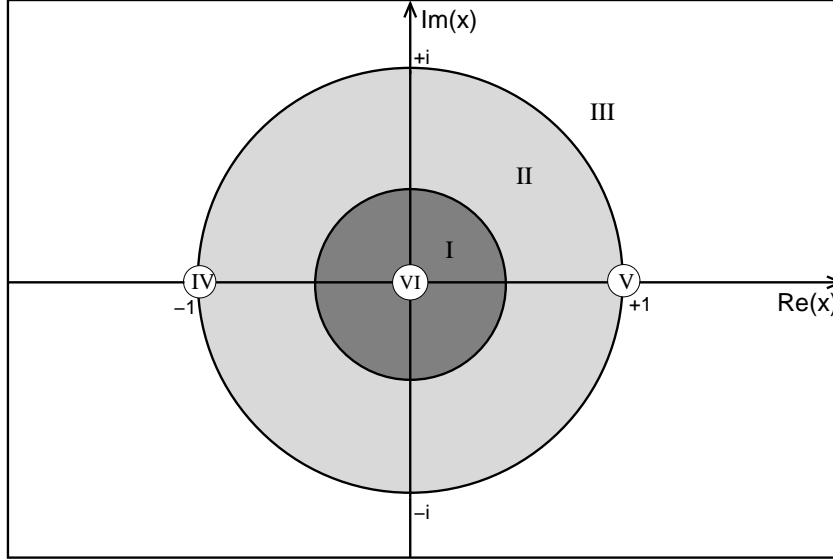


Fig. 1. The different regions of the complex plane used inside the CHAPLIN library. value of the argument z of the HPL. More precisely, the complex plane is divided into six regions (See Fig. 1),

- Region I: inside an annulus $0.025 < |z| \leq 0.3$, the basis functions are evaluated by using the expansions in $\log(1 - z)$ presented in Section 4.
- Region II: inside an annulus $0.3 \leq |z| \leq 1$, the basis functions are evaluated by using the expansions in $\log z$ presented in Section 4.
- Region III: points outside the unit disc, $|z| > 1$, are mapped back to the interior of the unit disc via inversion relations.
- Regions IV & V: The basis of Section 3 involves functions that are logarithmically divergent at ± 1 , leading to spurious singularities in the basis expansion. To avoid numerical instabilities caused by these spurious singularities, we use Taylor expansions close to $z = \pm 1$ to evaluate the individual HPL's without proceeding to a decomposition into basis functions.
- Region VI: In order to achieve a good numerical precision close to the origin of the complex plane, we use Taylor expansions in a disc $|z| < 0.025$ without proceeding to a decomposition into basis functions.

At the end of this procedure, CHAPLIN returns the numerical value of the HPL given as an input. In case the numerical evaluation of a divergent quantity is attempted (*e.g.*, $H(\vec{0}_n; 0)$ or $H(\pm 1, \vec{a}; \pm 1)$) an exception is thrown and the evaluation is aborted. Note that for real values of the argument, CHAPLIN uses the $' + i\varepsilon '$ prescription, *i.e.*, for $z \in \mathbb{R}$, $H(\vec{a}; z)$ is interpreted as $H(\vec{a}; z + i\varepsilon)$.

We conclude this section by giving an example of a sample program that prints all nine HPL's of weight two at the point $z = 1.54 + 0.91i$.

```

program chaplintest

double complex HPL2,z
integer n1,n2

z = dcmplx(1.54d0,0.91d0)
do n1=-1,1
  do n2=-1,1
    print*, HPL2(n1,n2,z)
  enddo
enddo

end program

```

5.3 Validation and comparison to other codes

We have compared the results obtained by CHAPLIN against **hplog** and GiNaC for real values of the argument and against HPL and GiNaC for complex values of the argument. The results for some sample points are summarized in Tab. 2 - 5. For real arguments, a small imaginary part is assumed. Note that the reduction of $H(1, -1, -1, 0; z)$ to basis functions is among the most complicated cases, involving almost 50 lines of **Fortran** code.

In order to give an idea of the CPU time needed per function call, we present in Tab. 1 the time for one million function calls on a 3 GHz Intel X5450 Processor for some HPL's in the six different regimes of the complex plane shown in Fig. 1. The running time varies only marginally inside a given region. The resulting average times for a single function call are given in Tab. 1 in units of microseconds (μs).

Region	I	II	III	IV	V	IV
$Li_2(z)$	1.4	4.4	4.5	0.2	0.2	0.3
$Li_3(z)$	1.9	4.7	5.0	0.2	0.2	0.3
$Li_4(z)$	6.2	8.9	9.1	4.2	4.3	4.5
$H(0, 1, 0, -1; z)$	8.7	17.3	44.2	4.3	4.4	4.5
$H(1, -1, -1, 0; z)$	140.0	213.4	393.8	4.3	4.8	4.7

Table 1

Average time per function call in microseconds (μs), for the six regions of the complex plane shown in Fig. 1.

$z = 0.5$		
$\text{Li}_2(z)$	CHAPLIN	5.8224052646501201e-01 + 0. <i>i</i>
	hplog	5.8224052646501256e-01 + 0. <i>i</i>
	GiNaC	5.8224052646501245e-01 + 0. <i>i</i>
$\text{Li}_3(z)$	CHAPLIN	5.3721319360804021e-01 + 0. <i>i</i>
	hplog	5.3721319360804010e-01 + 0. <i>i</i>
	GiNaC	5.3721319360804021e-01 + 0. <i>i</i>
$\text{Li}_4(z)$	CHAPLIN	5.1747906167389901e-01 + 0. <i>i</i>
	hplog	5.1747906167389945e-01 + 0. <i>i</i>
	GiNaC	5.1747906167389934e-01 + 0. <i>i</i>
$\text{H}(0, 1, 0, -1; z)$	CHAPLIN	7.7856141848313908e-02 + 0. <i>i</i>
	hplog	7.7856141848313090e-02 + 0. <i>i</i>
	GiNaC	7.7856141848313215e-02 + 0. <i>i</i>
$\text{H}(1, -1, -1, 0; z)$	CHAPLIN	-6.3908284909225732e-02 + 0. <i>i</i>
	hplog	-6.3908284909226009e-02 + 0. <i>i</i>
	GiNaC	-6.3908284909226135e-02 + 0. <i>i</i>

Table 2

Comparison between CHAPLIN, hplog and GiNaC for the point $z = 0.5$.

6 Summary

In this paper we have presented CHAPLIN, a new **Fortran** library to compute harmonic polylogarithms up to weight four for arbitrary complex argument. The algorithm is based on a reduction of HPL's to a set of basis functions which are then evaluated numerically using series expansions allowing for a very fast numerical convergence, hence rendering the computational cost of a function call quite modest. We have checked our numerical results against well-established codes [30,31,32,33] and found agreement to at least 14 digits for any argument in the complex plane.

Acknowledgements

The authors are grateful to B. Anastasiou, H. Gangl, T. Gehrmann and J. Rhodes for valuable discussions and comments. This work was supported by the Research Executive Agency (REA) of the European Union under the Grant Agreement number PITN-GA-2010-264564 (LHCPhenoNet) and by the Swiss National Foundation under contract SNF 200020-126632.

$z = 2.0$		
$\text{Li}_2(z)$	CHAPLIN	$2.4674011002723404\text{e}+00 + 2.1775860903036022\text{e}+00 \ i$
	hplog	$2.4674011002723399\text{e}+00 + 2.1775860903036017\text{e}+00 \ i$
	GiNaC	$2.4674011002723395\text{e}+00 - 2.1775860903036022\text{e}+00 \ i$
$\text{Li}_3(z)$	CHAPLIN	$2.7620719062289245\text{e}+00 + 7.5469382946024677\text{e}-01 \ i$
	hplog	$2.7620719062289241\text{e}+00 + 7.5469382946024799\text{e}-01 \ i$
	GiNaC	$2.7620719062289241\text{e}+00 - 7.5469382946024810\text{e}-01 \ i$
$\text{Li}_4(z)$	CHAPLIN	$2.4278628067547032\text{e}+00 + 1.7437130002545320\text{e}-01 \ i$
	hplog	$2.4278628067547032\text{e}+00 + 1.7437130002545298\text{e}-01 \ i$
	GiNaC	$2.4278628067547032\text{e}+00 - 1.7437130002545306\text{e}-01 \ i$
$\text{H}(0, 1, 0, -1; z)$	CHAPLIN	$5.1994752047739468\text{e}-01 + 1.7909927717176168\text{e}+00 \ i$
	hplog	$5.1994752047739512\text{e}-01 + 1.7909927717176164\text{e}+00 \ i$
	GiNaC	$5.1994752047739445\text{e}-01 - 1.7909927717176168\text{e}+00 \ i$
$\text{H}(1, -1, -1, 0; z)$	CHAPLIN	$8.0548200591357266\text{e}-01 - 1.3189461296972327\text{e}+00 \ i$
	hplog	$8.0548200591356789\text{e}-01 - 1.3189461296972333\text{e}+00 \ i$
	GiNaC	$8.0548200591356811\text{e}-01 + 1.3189461296972318\text{e}+00 \ i$

Table 3

Comparison between CHAPLIN, hplog and GiNaC for the point $z = 2.0$. Note that GiNaC uses a different convention for the imaginary parts.

$z = 0.5 + 0.5i$		
$\text{Li}_2(z)$	CHAPLIN	4.5398526915029508e-01 + 6.4376733288926902e-01 i
	HPL	4.5398526915029541e-01 + 6.4376733288926879e-01 i
	GiNaC	4.5398526915029558e-01 + 6.4376733288926880e-01 i
$\text{Li}_3(z)$	CHAPLIN	4.8615953708555987e-01 + 5.7007740708876864e-01 i
	HPL	4.8615953708556014e-01 + 5.7007740708876930e-01 i
	GiNaC	4.8615953708556009e-01 + 5.7007740708876897e-01 i
$\text{Li}_4(z)$	CHAPLIN	4.9578112182183881e-01 + 5.3402238407975344e-01 i
	HPL	4.9578112182183897e-01 + 5.3402238407975377e-01 i
	GiNaC	4.9578112182183876e-01 + 5.3402238407975355e-01 i
$\text{H}(0, 1, 0, -1; z)$	CHAPLIN	-3.6325772179994109e-02 + 1.384991682646747e-01 i
	HPL	-3.6325772179994879e-02 + 1.3849916826467456e-01 i
	GiNaC	-3.6325772179994845e-02 + 1.3849916826467457e-01 i
$\text{H}(1, -1, -1, 0; z)$	CHAPLIN	9.1142643382278842e-02 - 9.8191320890700317e-02 i
	HPL	9.1142643382278176e-02 - 9.8191320890700678e-02 i
	GiNaC	9.1142643382278163e-02 - 9.8191320890700595e-02 i

Table 4

Comparison between CHAPLIN, HPL and GiNaC for the point $z = 0.5 + 0.5i$.

A Proof of the associativity of the convolution product

In this appendix we proof the associativity of the convolution product, Eq. (4.2). Using the definition of the convolution product, Eq. (4.1), we obtain,

$$((a * b) * c)_N = \sum_{m=0}^N \binom{N}{m} (a * b)_m c_{N-m} = \sum_{m=0}^N \sum_{n=0}^m \binom{N}{m} \binom{m}{n} a_n b_{m-n} c_{N-m}. \quad (\text{A.1})$$

We now exchange the sums over m and n and shift the summation variable such that all sums start from zero. This gives,

$$\begin{aligned} ((a * b) * c)_N &= \sum_{n=0}^N \sum_{m=n}^N \binom{N}{m} \binom{m}{n} a_n b_{m-n} c_{N-m} \\ &= \sum_{n=0}^N \sum_{m=0}^{N-n} \binom{N}{m+n} \binom{m+n}{n} a_n b_m c_{N-m-n} \\ &= \sum_{n=0}^N \sum_{m=0}^n \binom{N}{m+N-n} \binom{m+N-n}{N-n} a_{N-n} b_m c_{n-m}, \end{aligned} \quad (\text{A.2})$$

$z = 2.0 + 2.0 i$		
$\text{Li}_2(z)$	CHAPLIN	3.4497312626178323e-01 + 2.7342872186403562e+00 i
	HPL	3.4497312626178264e-01 + 2.7342872186403560e+00 i
	GiNaC	3.4497312626178261e-01 + 2.7342872186403562e+00 i
$\text{Li}_3(z)$	CHAPLIN	1.2370548907501702e+00 + 2.7024607822310069e+00 i
	HPL	1.2370548907501697e+00 + 2.7024607822310064e+00 i
	GiNaC	1.2370548907501697e+00 + 2.7024607822310065e+00 i
$\text{Li}_4(z)$	CHAPLIN	1.7008027579027265e+00 + 2.4625762177390942e+00 i
	HPL	1.7008027579027260e+00 + 2.4625762177390939e+00 i
	GiNaC	1.7008027579027261e+00 + 2.4625762177390937e+00 i
$\text{H}(0, 1, 0, -1; z)$	CHAPLIN	-1.3092921033357463e+00 + 8.6009513536901472e-01 i
	HPL	-1.3092921033357458e+00 + 8.6009513536901561e-01 i
	GiNaC	-1.3092921033357459e+00 + 8.6009513536901561e-01 i
$\text{H}(1, -1, -1, 0; z)$	CHAPLIN	1.3154184588794104e+00 - 2.6274818437873471e-01 i
	HPL	1.3154184588794054e+00 - 2.6274818437872689e-01 i
	GiNaC	1.3154184588794056e+00 - 2.6274818437872688e-01 i

Table 5

Comparison between CHAPLIN, HPL and GiNaC for the point $z = 2.0 + 2.0 i$.

where the last step follows from changing the summation variable according to $n \rightarrow N - n$. The product of binomials can be simplified,

$$\begin{aligned}
\binom{N}{m+N-n} \binom{m+N-n}{N-n} &= \frac{N!}{(N+m-n)!(n-m)!} \frac{(N+m-n)!}{m!(N-n)!} \\
&= \frac{N!}{n!(N-n)} \frac{n!}{m!(n-m)!} = \binom{N}{n} \binom{n}{m},
\end{aligned} \tag{A.3}$$

yielding,

$$((a * b) * c)_N = \sum_{n=0}^N \sum_{m=0}^n \binom{N}{n} \binom{n}{m} a_{N-n} b_m c_{n-m} = (a * (b * c))_N. \tag{A.4}$$

B Derivation of Eq. (4.34)

In this appendix we present the derivation of the series expansions given in Eq. (4.34). The derivations follow the same spirit as in all other cases discussed in Section 4, *i.e.*, we start from the integral representation and perform a

change of variable, before inserting the Taylor expansions of the integrand. The proof involves some technical issues, which are discussed in detail in the following on the example of $H(0, 1, 0, -1; e^x)$. All other cases are similar.

To derive the Taylor expansion of $H(0, 1, 0, -1; e^x)$, we start from the series expansion of $H(-1; e^x) = \log(1 + e^x)$ and then integrate up to $H(0, 1, 0, -1; e^x)$. The series expansion of $H(-1; e^x) = \log(1 + e^x)$ is easily obtained from the integral representation and the generating function of the Genocchi numbers, Eq. (4.13). Letting $t = e^{t'}$, we get

$$\begin{aligned} H(-1; e^x) &= H(-1; 1) + \int_1^{e^x} \frac{dt}{1+t} = \log 2 - \frac{1}{2} \int_0^x \frac{dt'}{t'} \frac{2(-t')}{1+e^{-t'}} \\ &= \log 2 - \frac{1}{2} \sum_{n=0}^{\infty} \frac{\overline{G}_n}{n!} \int_0^x dt' t'^{n-1} = \log 2 - \frac{1}{2} \sum_{n=0}^{\infty} \frac{\overline{G}_n}{n!} \frac{x^n}{n} \quad (\text{B.1}) \\ &= \log 2 - \frac{1}{2} \sum_{n=0}^{\infty} \frac{\gamma_n}{n!} x^n, \end{aligned}$$

where we introduced the shorthand

$$\gamma_n = \frac{\overline{G}_n}{n}. \quad (\text{B.2})$$

Next we determine the series expansion of $H(0, -1; e^x) = -\text{Li}_2(-e^x)$. Repeating exactly the same steps as for $H(-1; e^x)$, we obtain

$$\begin{aligned} H(0, -1; e^x) &= -\text{Li}_2(-1) + \int_1^{e^x} \frac{dt}{t} H(-1; t) = \frac{\pi^2}{12} + \int_0^x dt' H(-1; e^{t'}) \\ &= \frac{\pi^2}{12} + \log 2 x - \frac{1}{2} \sum_{n=0}^{\infty} \frac{\gamma_n}{(n+1)!} x^{n+1}. \quad (\text{B.3}) \end{aligned}$$

If we try to repeat the same procedure for $H(1, 0, -1; e^x)$, a technical difficulty arises: in the previous example we had the split the integral over $[0, e^x]$ into two contributions from $[0, 1]$ and $[1, e^x]$. In the present case, however, we cannot do this, because $H(1, 0, -1; 1)$ is divergent. We therefore introduce a regulator ε and split the integration region into $[0, e^\varepsilon]$ and $[e^\varepsilon, e^x]$, and we will take the

limit $\varepsilon \rightarrow 0$ at the very end. We then obtain,

$$\begin{aligned}
H(1, 0, -1; e^x) &= H(1, 0, -1; e^\varepsilon) + \int_{e^\varepsilon}^{e^x} \frac{dt}{1-t} H(0, -1; t) \\
&= H(1, 0, -1; e^\varepsilon) - \int_{\varepsilon}^x \frac{dt'}{t'} \frac{(-t')}{e^{-t'} - 1} H(0, -1; e^{t'}) \\
&= H(1, 0, -1; e^\varepsilon) + \frac{\pi^2}{12} [H(1; e^x) - H(1; e^\varepsilon)] \\
&\quad + \log 2 [H(1, 0; e^x) - H(1, 0; e^\varepsilon)] + \frac{1}{2} \sum_{n=0}^{\infty} \frac{(\overline{B} * \dot{\gamma})_n}{(n+1)!} x^{n+1}.
\end{aligned} \tag{B.4}$$

We now have to send the regulator to zero. Concentrating only on terms depending on ε and using the shuffle algebra for HPL's, we obtain,

$$\begin{aligned}
&H(1, 0, -1; e^\varepsilon) - \frac{\pi^2}{12} H(1; e^\varepsilon) - \log 2 H(1, 0; e^\varepsilon) \\
&= H(1; e^\varepsilon) H(0, -1; e^\varepsilon) - \frac{\pi^2}{12} H(1; e^\varepsilon) \\
&\quad - H(0, 1, -1; e^\varepsilon) - H(0, -1, 1; e^\varepsilon) - \log 2 H(1, 0; e^\varepsilon).
\end{aligned} \tag{B.5}$$

All the terms in the second line have a smooth limit as $\varepsilon \rightarrow 0$, and the two divergent terms in the first line cancel exactly. This leaves us with

$$\begin{aligned}
H(1, 0, -1; e^x) &= -\frac{5}{8} \zeta_3 - \frac{\pi^2}{12} \log 2 + \frac{\pi^2}{12} H(1; e^x) + \log 2 H(1, 0; e^x) \\
&\quad + \frac{1}{2} \sum_{n=0}^{\infty} \frac{(\overline{B} * \dot{\gamma})_n}{(n+1)!} x^{n+1}.
\end{aligned} \tag{B.6}$$

The last integration is easy to perform and we immediately obtain

$$\begin{aligned}
H(0, 1, 0, -1; e^x) &= -4 \operatorname{Li}_4\left(\frac{1}{2}\right) - \frac{5}{2} \zeta_3 \log 2 + \frac{17\pi^4}{480} - \frac{1}{6} \log^4 2 \\
&\quad + \frac{\pi^2}{6} \log^2 2 - \frac{5}{8} \zeta_3 x + \frac{\pi^2}{6} \log 2 x + \frac{\pi^2}{12} \operatorname{Li}_2(e^x) \\
&\quad + \log 2 x \operatorname{Li}_2(e^x) - 2 \log 2 \operatorname{Li}_3(e^x) + \frac{1}{2} \sum_{n=0}^{\infty} \frac{(\overline{B} * \dot{\gamma})_n}{(n+2)!} x^{n+2},
\end{aligned} \tag{B.7}$$

which agrees with Eq. (4.34) after replacing $x = \log z$.

References

- [1] N. Nielsen, “Der Eulersche Dilogarithmus und seine Verallgemeinerungen,” Nova Acta Leopoldina (Halle) 90 (1909) 123.

- [2] A. B. Goncharov, “Multiple polylogarithms, cyclotomy and modular complexes,” *Math. Research Letters* **5** No. 4 (1998).
- [3] A. B. Goncharov, “Multiple polylogarithms and mixed Tate motives,” (2001) [math/0103059].
- [4] E. Remiddi and J. A. M. Vermaseren, “Harmonic polylogarithms,” *Int. J. Mod. Phys. A* **15**, 725 (2000) [hep-ph/9905237].
- [5] T. Gehrmann and E. Remiddi, “Two loop master integrals for $\gamma^* \rightarrow 3$ jets: The Planar topologies,” *Nucl. Phys. B* **601** (2001) 248, [hep-ph/0008287].
- [6] J. Ablinger, J. Blumlein and C. Schneider, “Harmonic Sums and Polylogarithms Generated by Cyclotomic Polynomials,” [arXiv:1105.6063].
- [7] J. A. M. Vermaseren, A. Vogt and S. Moch, “The Third-order QCD corrections to deep-inelastic scattering by photon exchange,” *Nucl. Phys. B* **724** (2005) 3 [hep-ph/0504242].
- [8] S. Moch, J. A. M. Vermaseren and A. Vogt, “The Longitudinal structure function at the third order,” *Phys. Lett. B* **606** (2005) 123 [hep-ph/0411112].
- [9] A. Vogt, S. Moch and J. A. M. Vermaseren, “The Three-loop splitting functions in QCD: The Singlet case,” *Nucl. Phys. B* **691** (2004) 129 [hep-ph/0404111].
- [10] S. Moch, J. A. M. Vermaseren and A. Vogt, “The Three loop splitting functions in QCD: The Nonsinglet case,” *Nucl. Phys. B* **688** (2004) 101 [hep-ph/0403192].
- [11] R. Bonciani, P. Mastrolia and E. Remiddi, “Master integrals for the two loop QCD virtual corrections to the forward backward asymmetry,” *Nucl. Phys. B* **690** (2004) 138 [hep-ph/0311145].
- [12] W. Bernreuther, R. Bonciani, T. Gehrmann, R. Heinesch, T. Leineweber, P. Mastrolia and E. Remiddi, “Two-loop QCD corrections to the heavy quark form-factors: The Vector contributions,” *Nucl. Phys. B* **706** (2005) 245 [hep-ph/0406046].
- [13] W. Bernreuther, R. Bonciani, T. Gehrmann, R. Heinesch, T. Leineweber, P. Mastrolia and E. Remiddi, “Two-loop QCD corrections to the heavy quark form-factors: Axial vector contributions,” *Nucl. Phys. B* **712** (2005) 229 [hep-ph/0412259].
- [14] W. Bernreuther, R. Bonciani, T. Gehrmann, R. Heinesch, T. Leineweber, E. Remiddi, “Two-loop QCD corrections to the heavy quark form-factors: Anomaly contributions,” *Nucl. Phys. B* **723** (2005) 91-116. [hep-ph/0504190].
- [15] P. Mastrolia and E. Remiddi, “Two loop form-factors in QED,” *Nucl. Phys. B* **664** (2003) 341 [hep-ph/0302162].
- [16] R. Bonciani, A. Ferroglia, P. Mastrolia, E. Remiddi and J. J. van der Bij, “Two-loop $N(F) = 1$ QED Bhabha scattering: Soft emission and numerical evaluation of the differential cross-section,” *Nucl. Phys. B* **716** (2005) 280 [hep-ph/0411321].

- [17] R. Bonciani, A. Ferroglia, P. Mastrolia, E. Remiddi and J. J. van der Bij, “Two-loop $N(F)=1$ QED Bhabha scattering differential cross section,” Nucl. Phys. B **701** (2004) 121 [hep-ph/0405275].
- [18] M. Czakon, J. Gluza and T. Riemann, “Master integrals for massive two-loop bhabha scattering in QED,” Phys. Rev. D **71** (2005) 073009 [hep-ph/0412164].
- [19] Z. Bern, M. Czakon, L. J. Dixon, D. A. Kosower and V. A. Smirnov, “The Four-Loop Planar Amplitude and Cusp Anomalous Dimension in Maximally Supersymmetric Yang-Mills Theory,” Phys. Rev. D **75** (2007) 085010 [hep-th/0610248].
- [20] G. Heinrich and V. A. Smirnov, “Analytical evaluation of dimensionally regularized massive on-shell double boxes,” Phys. Lett. B **598** (2004) 55 [hep-ph/0406053].
- [21] V. A. Smirnov, “Analytical result for dimensionally regularized massive on-shell planar double box,” Phys. Lett. B **524** (2002) 129 [hep-ph/0111160].
- [22] L. V. Bork, D. I. Kazakov and G. S. Vartanov, “On form factors in $N=4$ sym,” JHEP **1102** (2011) 063 [arXiv:1011.2440].
- [23] V. Del Duca, C. Duhr and V. A. Smirnov, “The Two-Loop Hexagon Wilson Loop in $N = 4$ SYM,” JHEP **1005** (2010) 084 [arXiv:1003.1702].
- [24] J. M. Henn, S. G. Naculich, H. J. Schnitzer and M. Spradlin, “More loops and legs in Higgs-regulated $N=4$ SYM amplitudes,” JHEP **1008** (2010) 002 [arXiv:1004.5381].
- [25] U. Aglietti, R. Bonciani, G. Degrassi and A. Vicini, “Analytic Results for Virtual QCD Corrections to Higgs Production and Decay,” JHEP **0701** (2007) 021 [hep-ph/0611266].
- [26] U. Aglietti, R. Bonciani, G. Degrassi and A. Vicini, “Master integrals for the two-loop light fermion contributions to $gg \rightarrow H$ and $H \rightarrow \gamma\gamma$,” Phys. Lett. B **600** (2004) 57 [hep-ph/0407162].
- [27] U. Aglietti, R. Bonciani, G. Degrassi and A. Vicini, “Two loop light fermion contribution to Higgs production and decays,” Phys. Lett. B **595** (2004) 432 [hep-ph/0404071].
- [28] T. Gehrmann and E. Remiddi, “Two loop master integrals for $\gamma^* \rightarrow 3$ jets: The Nonplanar topologies,” Nucl. Phys. B **601** (2001) 287 [hep-ph/0101124].
- [29] C. Anastasiou, S. Beerli, S. Bucherer, A. Daleo and Z. Kunszt, “Two-loop amplitudes and master integrals for the production of a Higgs boson via a massive quark and a scalar-quark loop,” JHEP **0701** (2007) 082 [hep-ph/0611236].
- [30] T. Gehrmann and E. Remiddi, “Numerical evaluation of harmonic polylogarithms,” Comput. Phys. Commun. **141**, 296 (2001) [hep-ph/0107173].

- [31] D. Maitre, “HPL, a Mathematica implementation of the harmonic polylogarithms,” *Comput. Phys. Commun.* **174**, 222 (2006) [hep-ph/0507152].
- [32] D. Maitre, “Extension of HPL to complex arguments,” [hep-ph/0703052].
- [33] J. Vollinga and S. Weinzierl, “Numerical evaluation of multiple polylogarithms,” *Comput. Phys. Commun.* **167**, 177 (2005) [hep-ph/0410259].
- [34] A. B. Goncharov, “A simple construction of Grassmannian polylogarithms,” [arXiv:0908.2238].
- [35] H. Cohen, L. Lewin and D. Zagier, “A Sixteenth-order Polylogarithm Ladder,” *Experimental Mathematics*, Vol. 1, No. 1, 1992, pp. 25-34.
- [36] C. Duhr, H. Gangl and J. Rhodes, “Symbol calculus for polylogarithms and Feynman integrals,” *in preparation*.
- [37] H. Gangl, “Regulators via iterated integrals (numerical computations),” *Clay Mathematics Proceedings*, Vol. 12, 2009.